DMAH@VLDB 2019

Los Angeles, CA, USA

Comparison of Approaches for Querying Chemical Compounds

Vojtěch Šípek, Irena Holubová, Martin Svoboda
svoboda@ksi.mff.cuni.cz

August 30, 2019

Charles University, Faculty of Mathematics and Physics
Prague, Czech Republic

Introduction

Chemical database

- Set of chemical compounds
  - Even up to 100 million molecules
- Each modeled as a **graph**
  - With **specific features** $\rightarrow$ their utilization

Existing solutions

- Storing and **querying**
- Various efficiency
  - Existing comparisons have several shortcomings

$\rightarrow$ **Unbiased comparison**

- Implementation of selected approaches
- Their comparison using a proposed **benchmark**

Chemical Compounds

Chemical compound = (simple) **undirected labeled graph**

- Set of **vertices**
  - Representing individual **atoms**, labeled with their kind
    - Carbon, oxygen, hydrogen, …
- Set of **edges**
  - Representing **chemical bonds**, also labeled
    - Single, double, triple, …

Specific features

- **Sparse** and **connected**
- **Small labeling alphabets**
  - Less than 10 for edges, low hundreds for vertices
- **Sizes are variable**
  - Just several vertices up to hundreds (millions) of vertices

Chemical Databases

→ Querying in chemical databases is a **challenging task**

- Because of the size and number of graphs

Various **forms of querying**

- Shortest paths search
- Exact match querying
- Similarity search
- **Subgraph querying** (substructure search)
    - The most common means
        - In chemoinformatics, bioinformatics, pharmaceutic industry…
    - Our only interest

Subgraph Querying

Basic principle

- Obtain a list of graphs from the database that match the provided graph query pattern, i.e. contain it as a subgraph

Naive approach

- For every single data graph…
- … perform **graph isomorphism test**
  - Several algorithms: **Ullmann**, **VF2**, QuickSI, …
  - NP-complete

Heuristic optimizations

- Construction of a **candidate set** based on the available **index**
  - $\rightarrow$ number of required isomorphism tests is reduced
  - $\rightarrow$ overall execution time is reduced

Available Solutions

Indexing techniques

- **GraphGrepSX**, **GString**, **GIRAS**, GIndex, C-tree, GDIndex, …
  - Just a selection of the best performing methods

Commercial solutions

- Project AMBIT, JChem and ABCD Oracle cartridges
  - Implementation not always publicly available

Generic databases

- **Relational** or **graph** databases

Existing Comparisons

Experimental comparisons of **indexing techniques**

- Yes, they exist…
- … however, they were **created by authors** of these methods themselves
- … and there are several other drawbacks
  - Not all the **approaches** were always covered
  - Not all interesting **characteristics** were always measured
  - Different **data and queries** were used
  - Not clear which **parts of the datasets** were actually used
  - Unknown **graph isomorphism** algorithm
  - Unknown **implementation details** and applied optimizations
  - Not always consistent **conclusions**

$\rightarrow$ it makes sense to perform an **independent comparison**

Objectives and Contributions

Considered approaches

- GraphGrepSX, GString, GIRAS
    - Only GIRAS implementation acquired from its authors
    - In case of the others: missing implementation details
- Relational database (Oracle)
- Graph database (PGX)
    - Actually an in-memory analytic tool, not a database

Objectives

- **Implementation** (in Java)
- **Benchmark** proposal
- **Experimental evaluation**
    - Confirmation or disproof of several **hypotheses**
        - Since direct quantitative comparison would not be entirely fair

GraphGrepSX

Principle

- For a given chemical compound (graph) to be indexed…
  - For each present **label-path**…
    - i.e. concatenation of interleaved vertex / edge labels on a path
  - … **number of its occurrences** in a given graph is detected
- Only paths of length up to a parameterized limit are indexed
  - E.g. 6

Index structure

- **Suffix tree**
  - Based on all the available label-paths
  - Each node contains a set of (graph id, occurrence count) pairs

GString

Idea

- Naturally, (organic) chemical compounds consist of 3 types of semantic structures
    - **Paths, cycles, and stars**

Condensed graph

- Graph of a chemical compound is first transformed
    - **Detected structures** are collapsed and **replaced with special vertices**
- Other optimizations are also applied
    - Hydrogens are omitted (their number can be calculated)
    - Labels of carbons and single (saturated) bonds are omitted
- Unfortunately, wide range of unspecified details

GIRAS

Motivation

- Getting better pruning by **indexing specific features only**

Principle

- Try to find and **identify** certain **features** (subgraphs of chemical compounds) such that these features are **rare**…
  - I.e. at most a certain number of chemical compounds contain them as a subgraph
  - This number is called **graph support**
- **We start with graph support equal to 1**…
- … **and iteratively increase it**
  - Until all the chemical compounds are indexed

Graph Database

Query expression construction

- Straightforward, since the query language natively supports subgraph matching

Relational Database

Database schema

- Table **bonds** with 5 columns
  - Compound id, bond id, source / target atom ids, bond type

Query expression construction

- For a given graph query pattern…
- … its **minimal spanning tree** is found
  - Edge values correspond to the overall numbers of occurrences of such edges in the database (e.g. C–C)
  - Kruskal algorithm is used
- Starting with (any) edge with the minimal value and continuing via BFS…
- … selection conditions are added for individual edges

Proposed Benchmark

Benchmark features

- **Data**
  - **ChEMBL** (release 24)
    - Manually curated database of bioactive molecules with drug-like properties
    - Almost 2 million compounds
  - Only the first **100,000 compounds** selected
    - In order to fit into the available system memory
    - Compounds with 1 to 548 atoms
    - 28 vertices and 30 edges on average
    - 18 vertex labels, 4 edge labels
- **Queries**
  - 4 sets of queries with 4, 8, 16, and 24 vertices respectively
  - Each set with 10 different query expressions

Performed Experiments

Environment

- Ordinary laptop
- 16 GB RAM
- Windows 10

Considered indicators (when applicable)

- **Index creation time**
- **Index and data size** (memory usage)
- **Candidate set calculation time**
- **Verification time** (graph isomorphism tests)
- **Overall query evaluation time**
- **Candidate set hit ratio**

Main Observations

GString

- **Condensed graphs** do not cause the index structure to be smaller
  - I.e. the number of indexed paths is even higher than in the original graphs

GIRAS

- **Index construction** is very slow
  - No result after 2 days even for just 10,000 compounds
  - Several hours needed for just hundreds of compounds
- **Indexing is not complete** and **not always works correctly**
  - I.e. we constructed a particular database and query which was not evaluated correctly

Main Observations

Indexing approaches in general

- **Candidate set calculation** plays minor role in the overall query evaluation time
  - I.e. graph isomorphism tests are time-demanding
  - $\rightarrow$ **the more intensive pruning, the better**

Relational database

- Contrary to usual expectations, it is a viable solution

Overall winner = **GraphGrepSX**

- Simple to implement
- The best overall performance
- Reasonable index size as well as its construction time

Conclusion

- **Chemical databases**
- **Indexing approaches** and database systems
- **Independent comparison**
  - Benchmark
    - 100,000 chemical compounds from ChEMBL
    - 40 query expressions
  - Experimental evaluation
  - Observations
    - Some of the expected hypotheses were confirmed
    - Some disproved, on the contrary
    - Certain results are not completely valid
- **GraphGrepSX** is the overall winner

Thank you for your attention…